PAPER *Special Section on Information and Communication System Security*

# Building a Scalable Web Tracking Detection System: Implementation and the Empirical Study

**Yumehisa HAGA**†a), **Yuta TAKATA**††b), ***Nonmembers***, **Mitsuaki AKIYAMA**††c), ***and*** **Tatsuya MORI**†d), ***Members***

**SUMMARY** Web tracking is widely used as a means to track user's behavior on websites. While web tracking provides new opportunities of e-commerce, it also includes certain risks such as privacy infringement. Therefore, analyzing such risks in the wild Internet is meaningful to make the user's privacy transparent. This work aims to understand how the web tracking has been adopted to prominent websites. We also aim to understand their resilience to the ad-blocking techniques. Web tracking-enabled websites collect the information called the web browser fingerprints, which can be used to identify users. We develop a scalable system that can detect fingerprinting by using both dynamic and static analyses. If a tracking site makes use of many and strong fingerprints, the site is likely resilient to the ad-blocking techniques. We also analyze the connectivity of the third-party tracking sites, which are linked from multiple websites. The link analysis allows us to extract the group of associated tracking sites and understand how influential these sites are. Based on the analyses of 100,000 websites, we quantify the potential risks of the web tracking-enabled websites. We reveal that there are 226 websites that adopt fingerprints that cannot be detected with the most of off-the-shelf anti-tracking tools. We also reveal that a major, resilient third-party tracking site is linked to 50.0 % of the top-100,000 popular websites.

***key words:*** *web tracking, web browser fingerprint*

## 1. Introduction

In the rapidly evolving web industry, people's behavior is diversifying, including viewing videos and purchasing on shopping sites. Companies track users' behavior, analyze their personality and tastes, and use the obtained information for online advertising or marketing. This activity is generally called **"Web Tracking."** However, web tracking poses some risks. In particular, there is the risk of privacy violations by revealing user behavior without the user consent. In addition, web tracking is exploited for malicious purposes, whereby certain users are lured to malicious sites such as fishing sites or drive-by-download sites. In 2016, high-profile sites in US was hit by massive malvertising campaign [1]. Although some countermeasures against web tracking have been proposed, the tracking technology itself is increasingly sophisticated. Thus, there are actually no methods to entirely avoid tracking.

Manuscript received September 8, 2016.
Manuscript revised January 30, 2017.
Manuscript publicized May 18, 2017.

†The authors are with the Dept. of Communication Engineering, Waseda University, Tokyo, 169–8555 Japan.
††The authors are with the NTT Secure Platform Laboratories, Musashino-shi, 180–8585 Japan.
a) E-mail: yumehisa.haga@nsl.cs.waseda.ac.jp
b) E-mail: takata.yuta@lab.ntt.co.jp
c) E-mail: akiyama.mitsuaki@lab.ntt.co.jp
d) E-mail: mori@nsl.cs.waseda.ac.jp

DOI: 10.1587/transinf.2016ICP0020

Furthermore, there is still an open debate on whether all web tracking activities are malicious and should be disabled. Web tracking provides certain benefits to the users, such as allowing sites to display ads that are more relevant to the users' interests. Therefore, it is commonly accepted that users should determine whether web tracking should be allowed. The World Wide Web Consortium has made the "Do not track" function standard. By activating "Do not track," [2] users can automatically deny all tracking requests from websites and thus protect themselves from the risks of tracking. However, tracking is actually performed by most websites without restriction, irrespective of the will of the user. For example, in 2013, Facebook paid 9.5 million dollars after litigation on a matter of unauthorized use of users' online histories in the US [3].

### 1.1 Mothods of Web Tracking

When websites conduct web tracking, they need to identify the user. They often use cookies as identifiers. However, tracking with cookies is restrictive because a cookie can be easily deleted or blocked by a simple operation and cannot be accessed across domains owing to the same-origin policy. Therefore, as an alternative, an identifier called the web browser fingerprint (**WBF**) is recently used. The WBF combines several feature points such as the type of the user's browser, screen resolution, installed plug-in, and installed fonts. The WBF is easily collected with JavaScript. Eckersley et al. [4] indicated that in JavaScript and Flash-enabled browsers, websites can identify the user's browser with a 94.2% accuracy using the WBF. Tracking performance or "Tracking power" depends on the WBF combination. There are the following three primary differences between a WBF and a cookie: 1) There are no restrictions such as the same-origin policy, and the WBF can be accessed across domains, 2) It is difficult for the user to delete or modify the value of the WBF, and 3) It is difficult to determine the use of the WBF, normal or for tracking. Therefore, the WBF is a greater threat than the cookie in tracking. Thus, in the present study, we focus on web tracking using WBF.

### 1.2 First-Party Site and Third-Party Site

A first-party site is a website visited by the user, whereas a third-party site is an external website that is linked to the first-party site. Third-party sites are located in domains different from those of first-party sites. They are linked from many different websites at the same time and track users across the Internet. Therefore, third-party tracking is more influential than first-party tracking, and countermeasures are urgently required. Conversely, first-party sites sometimes acquire WBFs to adjust the appearance of web screen. Consequently, disabling WBFs would lead to a decrease in usability [5]. In this study, we focus on web tracking only by third-party sites, which are henceforth known as "Tracking Sites."

### 1.3 Purpose and Approach

Our study aims to proactively investigate web tracking and help establish measures for detecting or blocking web tracking. In particular, we focus on third-party tracking using WBFs. In particular, we calculate the "tracking ability" and the "influence score" of every tracking site and visualize the potential risk of web tracking. To achieve this, we develop a system that crawls websites and detects fingerprinting by tracking sites. This crawler detects tracking sites exhaustively using static and dynamic analysis with JavaScript.

### 1.4 Contribution

The major contribution of our work is outlined as follows:

- We develop a generic tracking detection system using JavaScript analysis.
- We propose methods that group related tracking sites and quantify the potential risk of web tracking using multiple scales.
- Using our system, we examine most major tracking sites, which are linked to half the popular websites. We determine that most anti-tracking tools cannot defend the WBF.

We first review the most important related works in Sect. 2. Next, Sect. 3 presents an overview of the web tracking detection system used for our analysis. In Sect. 4, we show the results of investigating and analyzing tracking sites. Section 5 discusses some limitations of our work. Finally, Sect. 6 presents the conclusions.

## 2. Related Work

In this section, we describe several works related to web tracking or the WBF.

Eckersley et al. [4] demonstrated the effectiveness of using WBFs in web tracking for the first time. They showed that we can identify users with a 94.2% accuracy using information such as the user agent, the HTTP accept header, the screen resolution, the time zone, the installed plug-in, and the installed fonts. Afterward, many studies were conducted on the subject, which revealed that we can use performance measurements [6], JavaScript engines [7], rendering engines [8], Cascading Style Sheets [9], and HTML5 canvas [10] as fingerprints for identifying users.

Acer et al. [11], [12] developed a system that detected fingerprinting by dynamic analysis of JavaScript and investigated web tacking. This system revealed that many third-party sites are tracking users using the WBF.

Iso et al. [13] implemented a system for collecting and analyzing the WBF. After about four months' analyses of the collected 1,767 fingerprints, they observed that some of their values changed over time. In addition, they reported which fingerprints contribute more in identifying users, by calculating the fingerprint's variation degrees (entropy) for each device.

In the present work, we develop a system that is more advanced than the one constructed by Acer's team and investigate web tracking. In addition, we quantitatively compute the potential risk of web tracking based on several scales including the WBF feature observed by the Eckersley and Iso groups.

## 3. Web Tracking Detection System

We develop a system, which can detect the process of acquiring WBFs by investigating a tracking site. In this section, we explain the features of this system.

### 3.1 Web Browser Fingerprint

In Table 1, we show a list of WBFs that can be obtained by websites using JavaScript. In addition, we indicate the degrees of fingerprint variation (Entropy) as High, Med, and Low, as well as the resistance against passage of time (Duration) as Long, Med, and Short. If the entropy is high, a website can identify users more accurately. If the duration is long, websites can track users for a longer time. These evaluations were heuristically decided based on our research and the studies by Eckersley et al. [4] and Iso et al. [13] For example, in the case of a user agent, the entropy is relatively high because it contains information on the browser type and version, whereas the duration is short because the browser version changes frequently because of upgrading. In contrast, the platform represents user's OS, and its entropy is low because the type of OS is restrictive but the duration is long because users rarely change their OS. There are several other methods for obtaining the WBF, apart from JavaScript, such as Flash or HTTP request header, but JavaScript is most frequently used because it is the easiest to use. Therefore, we target the JavaScript tracking code in our investigation.

### 3.2 Overview of the System

An overview of the developed system appears in Fig. 1. There are three main components in the system: a headless browser, a driver, and an analyzer.

**Headless Brower:** A headless browser is a browser without a GUI. It has basic browser functions, such as loading and drawing web pages and running JavaScript, and it is often used for frontend testing by web developers. The implemented functions for web tracking detection in this case are

**Table 1** WBFs which can be taken by JavaScript

| Property or Method | Description | Entropy | Duration |
|---|---|---|---|
| toDataURL(), getImageData() | Canvas image data | High | Short |
| navigator.appCodeName | Code name of the browser | Low | Long |
| navigator.appName | Name of the browser | Low | Long |
| navigator.appVersion | Version of the brower | High | Short |
| navigator.userAgent | User-Agent | High | Short |
| navigator.mimeTypes | MIME types | Med | Long |
| navigator.plugins | Installed plugins | High | Short |
| navigator.language | Language of the brower | Med | Long |
| navigator.platfoorm | Platfrom of the browser | Low | Long |
| screen.height, screen.width | Size of screen | Med | Med |
| screen.availHeight, screen.availWidth | Size of screen (excluding the Windows taskbar) | Med | Med |
| screen.colorDepth | Bit depth of the color palette | Med | Med |
| screen.pixelDepth | Color resolution of the screen | Med | Med |
| getFontList() | Installed fonts | High | Med |
| getTimezoneOffset() | Time Zone | Med | Med |

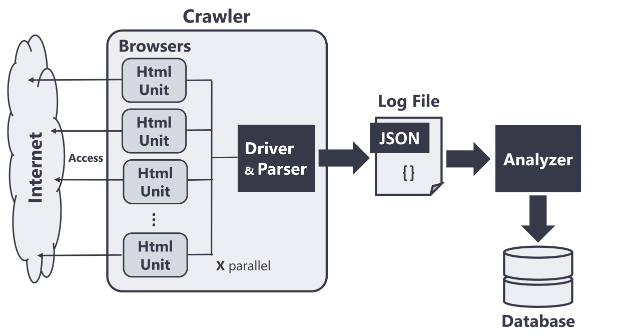

**Fig. 1** Overview of web tracking detection system

```
// Before
document.write("Hello, World");

// After
eval(function(p,a,c,k,e,d){e=function(c){return
c};if(!‘’.replace(/^/,String)){while(c--){d[c]
=k[c]||c}k=[function(e){return d[e]}];e=function()
{return‘¥¥w+’};c=1};while(c--){if(k[c]){p=
p.replace(new RegExp('¥¥b'+e(c)+'¥¥b','g'),
k[c])}}return p}('0.1("2, 3");',4,4,
'document|write|Hello|World'.split('|'),0,{}))
```

**Fig. 2** Example of JavaScript obfuscation

shown in Sects. 3.3 and 3.4. This headless browser can detect all WBF acquisition processes shown in Table 1. We used HtmlUnit [14] as a base headless browser.

**Driver:** The driver is a Python program, which runs headless browsers in parallel and parse from the output log to JSON files. A headless browser takes 3~10 seconds to analyze a website. Thus, we can increase the number of websites analyzed using the diver, depending on the machine specs, and crawl websites rapidly.

**Analyzer:** The analyzer conducts various analyses for the JSON files and stores the results in the database.

In this system, all these processes are automated, enabling the efficient investigation of websites.

## 3.3 Dynamic/Static Analysis of JavaScript

In our work, we expand the HtmlUnit mentioned above, and add functions for detecting fingerprinting by JavaScript. To obtain the WBF using JavaScript, it is necessary to access specific objects or run specific functions. For example, if we want to find the user agent, we should access "`navigator,userAgent`." Here, we detect fingerprinting through two approaches, namely, dynamic analysis and static analysis. In dynamic analysis, we run the JavaScript in the headless browser and hook access to objects or running functions related to the WBF. In static analysis, we dump JavaScript source codes and search the strings related to the WBF. The benefit of combining dynamic analysis and static analysis is significant. Dynamic analysis can evaluate functions running strictly and is not affected by the source code obfuscation, but it may overlook some instructions not running because of branch processes. We actually observed that tracking codes that perform branch processing depend on which WBF is taken. In contrast, static analysis cannot be used to evaluate functions running on strict mode, but is not affected by branch processing of JavaScript. Therefore, these two methods compensate each other's weakness, allowing the detection of fingerprinting more comprehensively.

## 3.4 Decode Obfuscation

In our system, we add a function to decode JavaScript obfuscation. Obfuscation is the process of encoding source codes so that people cannot read them. It is often used to protect against reverse engineering as well as conceal malicious or tracking codes. In JavaScript, the `eval()` function is primarily used for obfuscation (Fig. 2). If obfuscation is performed, we cannot apply static analysis. However, we can when we run the `eval()` function dynamically and dump its result. We implement this function on our system and enable performing static analysis of obfuscated JavaScript codes.

## 4. Investigation of Tracking Sites

In our study, we investigate tracking site using the system mentioned in Sect. 3. In this section, we describe the method of grouping tracking sites and quantifying the tracking risk, and we present the results of our analysis.

### 4.1 Target Websites

As we described in Sect. 1, tracking sites are external third-party websites, which are linked to first-party websites visited by users. Our target is tracking sites linked to the top one hundred thousand websites of the Alexa ranking [15]. We access these top pages and analyze JavaScript. We distinguish tracking sites by the domain (FQDN) of the URL, which we call the "Tracking Name." As a result of the crawls, we extract 37,812 tracking sites and determine that 17,290 of them are taking at least one WBF using JavaScript. The frequency of WBF use in the JavaScript files is presented in Fig.3. The most collected fingerprints are User-Agent, or screen sizes. These information is often referred when websites adjust the appearance of the screen.

### 4.2 Grouping Tracking Sites by Recursive Jaccard Calculation

Some tracking sites using different tracking names belong to the same community. These are likely to exchange tracking information. We call these tracking sites a "Tracking Group." If there is a big tracking group composed of small tracking sites, we may investigate each tracking site independently, without considering the tracking group as an entity. Thus, before analysis, we group the tracking sites.

We observed that tracking sites belonging to the same community are often linked to almost the same first-party sites. Therefore, we cluster tracking sites based on which first-party sites each tracking site is linked to. As a method to cluster tracking sites, we adopted an algorithm, which calculates the Jaccard coefficient recursively. The Jaccard coefficient is a measure of the similarity between two sets. First, we calculate the similarity between the two sets. When A is a set of first-party sites linked to one tracking site and B is a set of first-party sites linked to another tracking site, the Jaccard coefficient, which represents the similarity between A and B, is calculated as follows.

$$Jaccard = \frac{|A \cap B|}{|A \cup B|}$$

In our work, if the Jaccard coefficient is above 0.8, we regard these sets as the same tracking group. By applying this process for all tracking sites, there are some tracking groups that have a maximum of two elements. Next, we compare these tracking groups to form a bigger group (Fig. 4). Then, we repeat this process and the formed tracking groups increase each time. Finally, we finish this process, when no more new tracking groups are extracted. A similar method was used for Hostname-IS Cluster (HIC) in drive-by download research [16]. As a result of the recursive Jaccard calculation, tracking groups containing a maximum of three tracking sites are extracted, for example fb.travel-assets.com, a.travel-assets.com, c.travel-assets.comg, fwww.homeaway.jp, static0.homeaway.jp, static1.homeaway.jpg, fwww.tribdss.com, ssor.tribdss.com, www.trbas.comg. Since these tracking sites have similar names, we assume that they are connected.

### 4.3 Quantification of Tracking Risk

It is difficult for users to know how the WBF is utilized by the web administrator and whether the WBF is actually used for malicious purposes. However, it is possible to determine the "potential" risk suffered when users' tracking (privacy) information is leaked. To quantify such this risk of web tracking, we define two indexes, the Tracking Ability (TA) and the Influence Score (IS). TA represents the accuracy in user identification and the length of time that it is valid, and it is calculated from the entropy and the duration of the obtained WBF. If a website collects many high-entropy and long-duration WBFs, the tracking performance would be improved. When the set of collected WBFs is $X = \{x_1, x_2, \ldots, x_n\}$, TA is calculated as follows.

$$TA = \sum_{i=1}^{n} \left\{ \alpha E(x_i) + (1-\alpha) D(x_i) \right\}$$

E and D mean entropy and duration. We define the value of

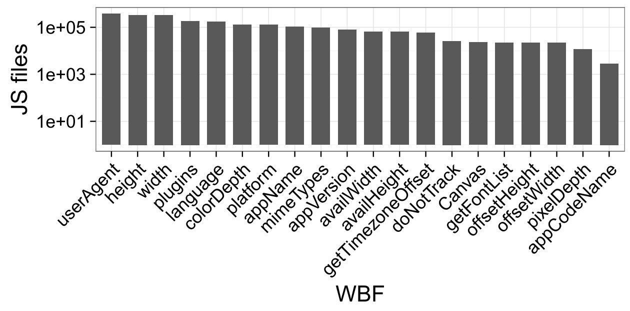

**Fig. 3** Frequency of use of WBF

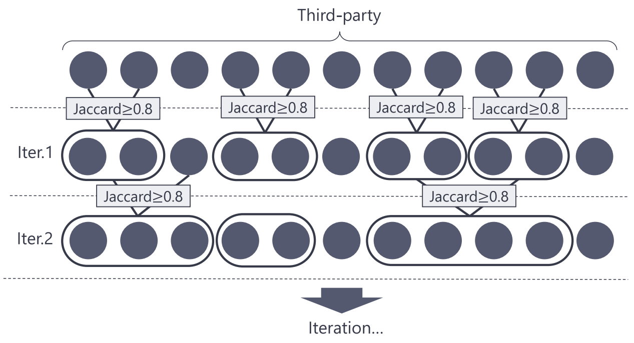

**Fig. 4** Recursive Jaccard calculation

entropy and duration as 3 pt, 2 pt, 1 pt based on the High, Med, Low and Long, Med, Short characterizations, respectively, as shown in Table 1. We empirically assign $\alpha = 0.75$ based on contribution of each metrics for user identification.

Conversely, influence score represents the range within which a tracking site can track a user, and it is calculated by the number of links from first-party sites and the number of visitors to each site. If the influence score is high, users can be tracked across a wide range of the Internet. The number of visitors to each website can be estimated from the Alexa ranking. When a set of first-party sites linked to a tracking site is $Y = \{y_1, y_2, \ldots, y_n\}$, IS is calculated as follows.

$$IS = \sum_{i=1}^{n} \frac{\max(R(y_i)) - R(y_i)}{\beta},$$

where R denotes the Alexa ranking and $\max(R(y_i)) = 100,000$, which is the number of Alexa ranked websites we studied. $\beta$ is the normalizing constant set to 1,000.

## 4.4 Result of Analysis

We present the distribution of TA and IS calculated using the method mentioned in Sect. 4.2. The tracking groups with high TA or IS are shown in Tables 2 and 3, respectively. All the tracking groups on these tables have only one tracking site.

According to Table 2, "cdn.krxd.net" has the highest TA (33.0 points), which means it takes almost all WBF on Table 1. "cdn.krxd.net" is obtained by Krux [17], which provides marketing services based on tracking users' information collected in websites. All tracking groups shown in Table 2 are organizations providing services such as marketing, advertisement, and information security based on access analysis.

Next, we evaluate whether the existing anti-tracking tools can block fingerprinting. We can measure the defensive performance of these tools by calculating TA based on the WBFs they can protect. If a tool has TA larger than the one computed for a website, it implies that the tool has the greater ability of protecting the tracking activities than the tracking ability of the website. It is worth noting that this does not necessarily indicate that the tool can protect all the tracking activities of websites with lower TAs because there could be several fingerprints that cannot be covered with the tool. However, since the tool with large TA can block majority of fingerprints for a given website, remaining fingerprints have less information to be used for tracking a user. Similarly, even a website has greater TA than a tool, that does not necessarily indicate that the website can always track all the users who are using the tool. Our objective here is to measure the performance of websites and tools with respect to their tracking ability and protecting ability, respectively. TA is a concise and useful metrics to understand the tracking ability of a broad spectrum of large-scale websites and the protecting ability of various existing tools. The cumulative IS represents the effectiveness of the anti-tracking tools. If a tool is highly effective, it can be used to protect from more tracking sites. In this study, we investigate the following anti-tracking tools: PriVaricator (PV) [18], FireGloves (FG) [19], Random Agent Spoofer (RAS) [20], Tor Browser (Tor) [21], and FP-Block (FPB) [5]. For each anti-tracking tool, we calculate TA, website coverage, and effectiveness on Table 4. Among these tools, FP-Block has the highest defensive performance. According to Table 2, FP-Block can protect from all tracking sites, whereas other tools do not have sufficient defensive performance for this. FP-Block is a relatively new tool, whereas other tools have been generally used for long time. This suggests that tracking technology

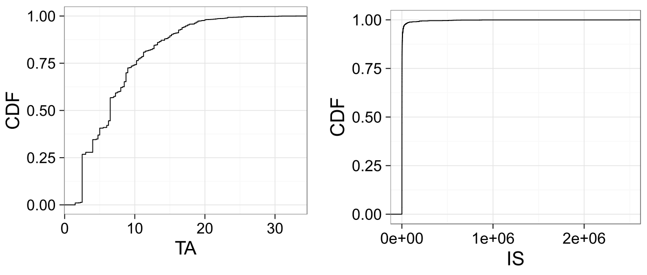

**Fig. 5** Distribution of TA (Left) and IS (Right)

**Table 2** High-TA tracking groups

| Third-party | Path | TA |
|---|---|---|
| cdn.krxd.net | ctjs/controltag.js.875f... | 33.0 |
| static.fraudmetrix.cn | fm.jsv?er=0.1&t=402217 | 31.0 |
| cse.google.com | adsense/search/async-ads.js | 30.5 |
| app.trustev.com | api/v2.0/TrustevJS?key=7a3... | 29.0 |
| cdn.tagcommander.com | 362/tc_Aspartam_3.js | 28.5 |
| tags.mdotlabs.com | tracking.php?siteID=e8AJ | 27.5 |
| static.audienceinsights.net | t.js | 26.0 |
| t.qservz.com | js/pi.js | 25.5 |
| tags.tiqcdn.com | utag/wsjdn/wsjpages/prod/ utag.56.js... | 25.5 |
| servedby.openxmarket.asia | servedby.openxmarket.asia/w/ 1.0/jstag | 25.5 |

**Table 3** High-IS tracking groups

| Third-party | Links | IS |
|---|---|---|
| www.google-analytics.com | 49,963 | 2,493,837 |
| pagead2.googlesyndication.com | 17,586 | 875,754 |
| connect.facebook.net | 17,147 | 853,935 |
| www.googletagmanager.com | 12,226 | 650,517 |
| partner.googleadservices.com | 10,241 | 599,434 |
| ajax.googleapis.com | 11,592 | 566,757 |
| www.googleadservices.com | 10,474 | 547,918 |
| apis.google.com | 10,310 | 512,459 |
| platform.twitter.com | 10,099 | 506,008 |
| tpc.googlesyndication.com | 9,715 | 449,372 |
| static.xx.fbcdn.net | 6,649 | 303,266 |

**Table 4** Defensive performance of anti-tracking tools

| Tool | TA | Coverage % | Effectivity |
|---|---|---|---|
| PriVaricator | 4.8 | 53.3 | 1,345,753 |
| FireGloves | 16.5 | 96.6 | 9,314,979 |
| Random Agent Spoofer | 20.0 | 98.7 | 14,354,210 |
| Tor Browser | 24.8 | 99.8 | 17,014,206 |
| FP-Block | 34.0 | 100.0 | 17,238,061 |

has been progressing recently. Indeed, Canvas fingerprint has been used in recent years [10], and old tools such as Fire-Gloves cannot protect against taking the Canvas fingerprint. Therefore, users who want to block tracking should choose an anti-tracking tool carefully, and, for now, using FP-Block is the best choice.

Next, according to Table 3, it is revealed that high IS tracking sites are organizations providing very popular services such as Google, Twitter, and Facebook. Surprisingly, Google Analytics [22], Google's access analysis service, has a much higher IS than others, and it is linked to half of the first-party sites investigated in this study. Nevertheless, Google Analytics' TA is 17.25 pt, and its tracking can be protected by using the anti-tracking tools mentioned above.

In Fig. 6, we show the information propagation which occurs, when we access Alexa top-ranked websites linked to Google Analytics, Facebook, or Twitter. The graph of Google Analytics, which has a high IS, has a large slope.

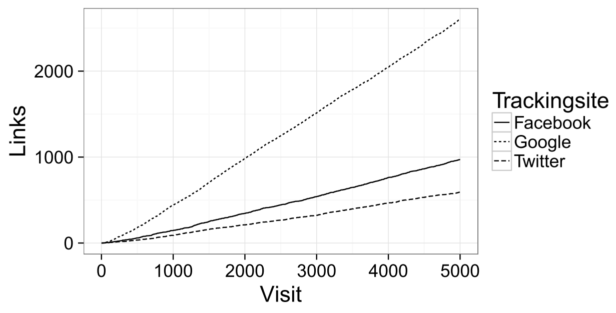

**Fig. 6** Information propagation on tracking site

Therefore, user information is leaked more rapidly than from other tracking sites. Consequently, it is important to consider third-party sites as well as first-party sites with regard to web tracking. In Fig. 7, we show the relationship between TA and IS for each tracking group. When both TA and IS are high (plot on the upper right), the web tracking risk is high. We can see Google, Facebook, Twitter, Yandex, Baidu, or AddThis on the upper right of the figure. And we can observe Google differences in collecting WBFs and adjustments in its TP depending on the services provided. The dashed line in this figure indicates the defensive performance of the anti-tracking tools. If a plot is above the line, the corresponding tracking site cannot be blocked using this tool. As we mentioned above, there are many tracking groups which cannot be blocked by anti-tracking tools except FP-Block.

## 5. Discussion

In this section, we discuss the limitations of our work and future work.

In JavaScript static analysis of our web tracking detection system, there are some false positives of fingerprinting. That is because we search strings related fingerprinting and some simple word such as "screen", "height" and "width" are often seen in the source code. So we need to reveal how much impact of false positives to the accuracy of web tracking detection, or take any measures to reduce false positives.

We defined entropy and duration and calculated the TP heuristically based on some researches, but we have not defined clear criteria for it. Therefore, it is necessary to im-

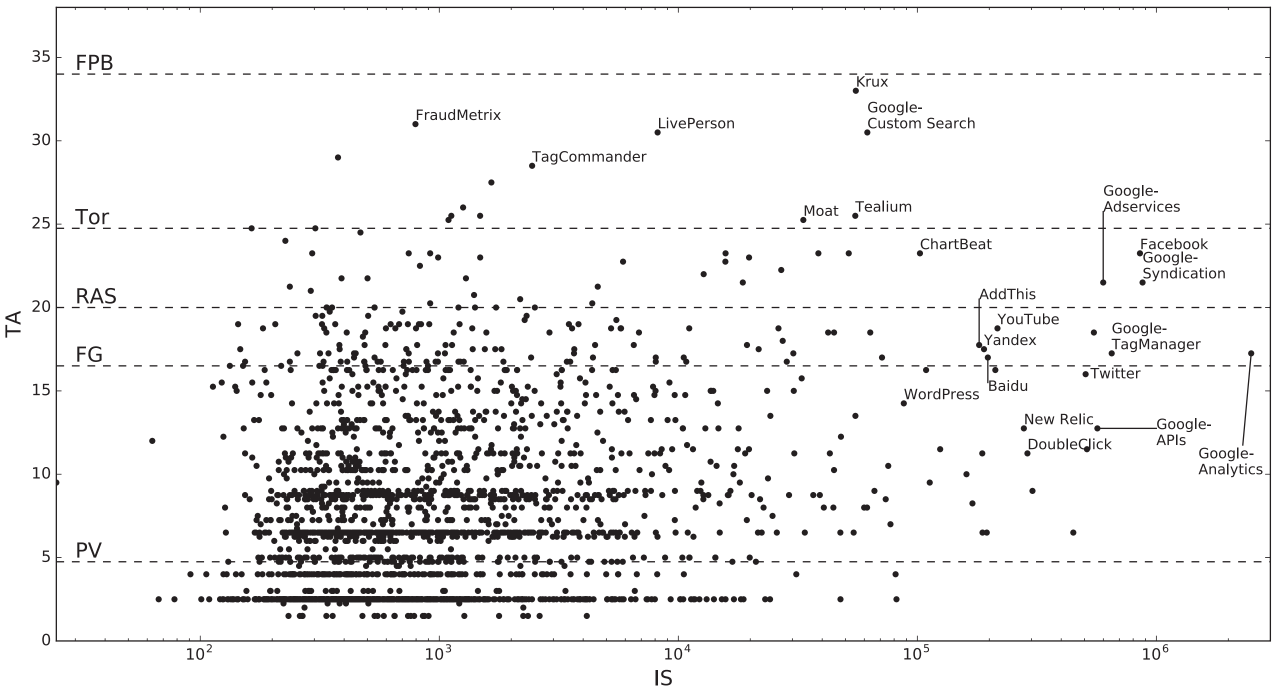

**Fig. 7** Distribution of tracking groups

plement a system, which collects and analyzes WBFs and defines the entropy and duration of WBFs quantitatively, to ensure the validity of these values.

Although we quantify the "potential" tracking risk, we cannot know how WBFs are actually utilized, and we cannot detect malicious tracking sites directly. Thus, for example, we could let a website take our own WFB, and if we identify any behaviors (such as advertisements) related to web tracking in different sites, we would know our information has been leaked unintentionally. In addition, when WBFs are taken even with the use of Do Not Track, the opt-out declaration of tracking, the website could be malicious.

In this work, we only focused on WBF, but our system we developed has the other several functions except WBF analysis. For example, the system can dump information such as Cookie, HTTP header, and asynchronous communication using Ajax. In addition, the headless browser can enable Do Not Track. These functions can be applied to various analysis related to web tracking in the future.

In addition, we plan to expand the range of investigation (0.1 million sites → 1 million sites) or change the crawling period. We expect we can find some new features of tracking group or relationship between Alexa ranking and tracking group.

## 6. Conclusion

We developed a scalable system that can detect third-party web tracking sites linked to many first-party sites. The key idea was to detect WBF-taking sites by combining dynamic and static analysis. Our extensive analysis using the Alexa top 100,000 sites revealed the many tracking sites that automatically collect WBFs from users. We aimed to quantify the power and influence of tracking sites and revealed that there are 226 of popular websites that adopt fingerprints that cannot be detected with the most of the off-the-shelf anti-tracking tools.

Our results indicated that web tracking sites with the powerful traking abilitiy cannot be blocked with the most of anti-tracking tools except the state-of-the-art tool, FP-block. These observations lead us to the following suggestions / future work for the two stakeholders; users who want to block tracking and tracking companies. First, we suggest users who want to block tracking to use the most powerful tracking detecting system such as FP-block. As we revealed, there are non-negligible number of web sites that use quite strong WBFs that cannot be detected with the existing tools. We note that major tracking sites are linked to the majority the popular websites; it indicates the ubiquity of the risk of tracking. We also note that even FP-block cannot detect all the tracking sites; this fact indicates the need for further research to detect tracking activities. Second, we advocate that tracking companies need to adopt more transparent tracking system so that users will not lose right to not be tracked. Since the purpose of performing tracking for the tracking company is to make revenue from the online services, which are freely available in many cases, we understand that it is always challenging to strike a good balance between privacy and healthy business model. One good approach toward the goal is to build a privacy-aware advertisement system. For instance, Guha et al. presented *Privad*, an online advertising system designed to more private than existing systems [23]. The Privad system has three main principles: users are not tracked, users opt-in, and advertisers cannot target sensitive information. In future, the tracking companies may want to adopt the system like Privad.

## References

[1] Trend Micro, "Massive Malvertising Campaign in US Leads to Angler Exploit Kit/BEDEP." http://blog.trendmicro.com/trendlabs-security-intelligence/malvertising-campaign-in-us-leads-to-angler-exploit-kitbedep/.
[2] "Tracking Preference Expression (DNT)." https://www.w3.org/TR/tracking-dnt/.
[3] D. Kravets, "Facebook's $9.5 Million 'Beacon' Settlement Approved." http://www.wired.com/2012/09/beacon-settlement-kapproved/.
[4] P. Eckersley, "How unique is your web browser?," Proc. 10th International Conference on Privacy Enhancing Technologies, PETS'10, pp.1–18, 2010.
[5] C.F. Torres, H. Jonker, and S. Mauw, "Fp-block: usable web privacy by controlling browser fingerprinting," Proc. 20th European Symposium on Research in Computer Security (ESORICS), Lect. Notes Comput. Sci., pp.3–19, 2015.
[6] K. Mowery, D. Bogenreif, S. Yilek, and H. Shacham, "Fingerprinting information in JavaScript implementations," Proc. Web 2.0 Workshop on Security and Privacy (W2SP) 2011, 2011.
[7] M. Schmiedecker, P. Reschl, M. Huber, M. Leithner, S. Schrittwieser, and E. Weippl, "Fast and reliable browser identification with JavaScript engine fingerprinting," Proc. Web 2.0 Workshop on Security and Privacy (W2SP) 2013, 2013.
[8] T. Unger, M. Mulazzani, D. Frühwirt, M. Huber, S. Schrittwieser, and E. Weippl, "SHPF: Enhancing HTTP(S) Session Security with Browser Fingerprinting," Proc. 2013 International Conference on Availability, Reliability and Security, ARES '13, pp.255–261, 2013.
[9] N. Takei, T. Saito, K. Takasu, and T. Yamada, "Web browser fingerprinting using only cascading style sheets," Proc. 10th International Conference on Broadband and Wireless Computing, Communication and Applications (BWCCA) 2015, pp.57–63, 2015.
[10] K. Mowery and H. Shacham, "Pixel perfect: Fingerprinting canvas in HTML5," Proc. Web 2.0 Workshop on Security and Privacy (W2SP) 2012, 2012.
[11] G. Acar, M. Juarez, N. Nikiforakis, C. Diaz, S. Gürses, F. Piessens, and B. Preneel, "Fpdetective: Dusting the web for fingerprinters," Proc. 2013 ACM SIGSAC Conference on Computer & Communications Security, CCS '13, pp.1129–1140, 2013.
[12] G. Acar, C. Eubank, S. Englehardt, M. Juarez, A. Narayanan, and C. Diaz, "The Web Never Forgets: Persistent Tracking Mechanisms in the Wild," Proc. 2014 ACM SIGSAC Conference on Computer and Communications Security, CCS '14, pp.674–689, 2014.
[13] Y. Iso, N. Kiryu, K. Tsukamoto, K. Takasu, T. Yamada, N. Takei, and T. Saito, "An implementation of browser fingerprinting website and analysis of its collected data (in japanese)," Proc. Computer Security Symposium (CSS) 2014, pp.377–370, 2014.
[14] "HtmlUnit." http://htmlunit.sourceforge.net/.
[15] Alexa Top Sites. http://www.alexa.com/topsites.
[16] J. Zhang, C. Seifert, J.W. Stokes, and W. Lee, "Arrow: Generating signatures to detect drive-by downloads," 2011.
[17] Krux. http://www.krux.com/.
[18] N. Nikiforakis, W. Joosen, and B. Livshits, "Privaricator: Deceiving fingerprinters with little white lies," Proc. 24th International Confer-

ence on World Wide Web, WWW '15, pp.820–830, 2015.
[19] K. Boda, Á.M. Földes, G.G. Gulyás, and S. Imre, "User tracking on the web via cross-browser fingerprinting," Proc. 16th Nordic Conference in Secure IT Systems, NordSec '11, pp.31–46, 2011.
[20] Random Agent Spoofer. https://github.com/jmealo/random-ua.js.
[21] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router.," Proc. 13th conference on USENIX Security Symposium - Volume 13, SSYM '04, pp.21–21, 2004.
[22] Google Analytics. https://www.google.com/analytics/.
[23] S. Guha, B. Cheng, and P. Francis, "Privad: practical privacy in online advertising," Proc. 8th USENIX conference on Networked systems design and implementation, NSDI '11, pp.169–182, 2011.

**Yumehisa Haga** was born in 1992. He recieved B.E. degree in computer science and engineering from Waseda University in 2015. He is currently a master course student in the Department of Computer Science and Communications Engineering, Waseda University. He has been engaged in research of network security and web security.

**Yuta Takata** received his B.E. and M.E. degrees in computer science and engineering from Waseda University in 2011 and 2013. He is currently a Ph.D. student in the Department of Computer Science and Communications Engineering, Waseda University. Since joining Nippon Telegraph and Telephone Corporation (NTT) in 2013, he has been engaged in research and development of network security, especially honeyclient and malicious code analysis. He is now with the Cyber Security Project of NTT Secure Platform Laboratories.

**Mitsuaki Akiyama** received the M.E. degree and Ph.D. degree in Information Science from Nara Institute of Science and Technology, Japan in 2007 and 2013, respectively. Since joining Nippon Telegraph and Telephone Corporation NTT in 2007, he has been engaged in research and development of network security, especially honeypot and malware analysis. He is now with the Network Security Project of NTT Secure Platform Laboratories.

**Tatsuya Mori** is currently an associate professor at Waseda University, Tokyo, Japan. He received B.E. and M.E. degrees in applied physics, and Ph.D. degree in information science from the Waseda University, in 1997, 1999 and 2005, respectively. He joined NTT lab in 1999. Since then, he has been engaged in the research of measurement and analysis of networks and cyber security. From Mar 2007 to Mar 2008, he was a visiting researcher at the University of Wisconsin-Madison. He received Telecom System Technology Award from TAF in 2010 and Best Paper Awards from IEICE and IEEE/ACM COMSNETS in 2009 and 2010, respectively. Dr. Mori is a member of ACM, IEEE, IEICE, and USENIX.